UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/643,574	08/18/2003	James R. Kohn	1376.730US1	3940

21186 7590 08/07/2007
SCHWEGMAN, LUNDBERG, WOESSNER & KLUTH, P.A.
P.O. BOX 2938
MINNEAPOLIS, MN 55402

EXAMINER	
FENNEMA, ROBERT E	

ART UNIT	PAPER NUMBER
2183	

MAIL DATE	DELIVERY MODE
08/07/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

MN

Office Action Summary	Application No. 10/643,574	Applicant(s) KOHN, JAMES R.	
	Examiner Robert E. Fennema	Art Unit 2183	

-- *The MAILING DATE of this communication appears on the cover sheet with the correspondence address* --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *12 July 2007*.
2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.
3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-34* is/are pending in the application.
4a) Of the above claim(s) _____ is/are withdrawn from consideration.
5)☒ Claim(s) *8* is/are allowed.
6)☒ Claim(s) *1-4,7,9-15,18,19,21-24 and 27-34* is/are rejected.
7)☒ Claim(s) *5,6,16,17,20,25 and 26* is/are objected to.
8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.
10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
a)☐ All b)☐ Some * c)☐ None of:
1.☐ Certified copies of the priority documents have been received.
2.☐ Certified copies of the priority documents have been received in Application No. _____.
3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *7/12/2007*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-34 have been considered.

Allowable Subject Matter

2. Claims 5-6, 16-17, 20, and 25-26 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

3. The following is a statement of reasons for the indication of allowable subject matter: While the applicant has disclosed in his admitted prior art that a sequence of values is written to and read back from memory to determine if duplicate addresses exist in a vector, the applicant has not disclosed in his admitted prior art how this sequence of values was determined in the conventional algorithms, nor has the method of determining as found in the applicants claims been found in any prior art.

4. Claim 8 is allowed.

5. The following is an examiner's statement of reasons for allowance: While the prior art has taught a method executing each of the steps as disclosed in the claim, the prior art has not disclosed performing the steps in a specific order, only that they occur.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

7. Claims 1-4, 7, 9-15, 18-19, 21-24, and 27-34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Beard et al. (USPN 5,640,524, herein Beard), in view of Applicants admitted prior art (herein Kohn), further in view of Cray Assembly Language Systems Reference Manual (herein Cray).

8. As per Claim 1, Beard teaches: A computerized method comprising:

providing a first vector of addressing values (Column 2 Line 65 – Column 3 Line 1);

providing a second vector of operand values (Column 3, Lines 10-12. The second vector instruction would necessarily need a vector of operand values in order to operate on the retrieved data words);

performing an arithmetic-logical operation using values from the third vector register and the compressed second vector of operand values to generate a result vector (Column 3, Lines 10-18); and

using the first vector of addressing values as masked by the bit vector, storing the result vector to memory (Column 3, Lines 13-18), but fails to teach:

storing a first sequence of values to a sequence of addressed locations within a constrained area of memory, wherein each location's address is based at least in part on a corresponding one of the addressing values;

reading back from the sequence of addressed locations values resulting from the storing of the first sequence to obtain a second sequence of values;

comparing the first sequence of values to the second sequence of values to generate a bit vector representing compares and miscompares.

However, Kohn has taught that in the prior art (the "conventional" algorithm as described on pages 4 and 5 of the specification) that after loading from memory, you would store a known pattern, load it back, and compare against the original pattern to determine collisions. Kohn further teaches that by doing this, not only would you see if there was a collision, but also which elements were colliding (Page 5). Given this disclosure, it would have been obvious to one of ordinary skill in the art at the time the invention was made to store a sequence of values to a constrained area of memory (which can be any writable area of memory for example, constrained can be read to be all of memory, or any arbitrary boundary of memory), reading it back out, and comparing the two to determine which elements did not compare correctly (and to do so, would require it to be stored somewhere), to allow correct execution of the instructions, which is critical for computers.

Beard further fails to teach:

compressing the second vector of operand values using the bit vector;

using the first vector of addressing values as masked by the bit vector, loading a third vector register with elements from memory.

However, Cray teaches an instruction set that runs vector instructions and supports a vector architecture, one of which would have been obvious to run on Beards invention, as Beard's invention is an improvement of the original Cray vector processor, as disclosed in Column 1 Line 66- Column 2, Line 6) and thus would be compatible, as every system needs some instruction set architecture, and Beard does not teach a specific instruction set to use in his invention. One of the instructions Cray teaches is a vector compress (Section 2.5.4), which uses a vector as a bit mask to modify a register to: invalidate certain elements, modify elements (if the destination is the same as source), and can also load a new register with elements (if the destination is different than the source), which is advantageous for improving the performance of decoupled execution, as also described in Section 2.5.4. Therefore, one of ordinary skill in the art at the time the invention was made would have recognized the improvement of decoupled execution of using mask and compression instructions, and would have modified Beards invention to include and utilize them to gain those performance advantages.

9. As per Claim 2, Beard teaches: The method of claim 1, wherein addresses of the elements in memory are calculated by adding each respective addressing value to a base address of an object in memory (Column 3 Lines 2-5).

10. As per Claim 3, Beard teaches: The method of claim 1, wherein the arithmetic-logical operation is an addition operation that produces at least one element of the result vector as a summation of an element of the loaded third vector register and a plurality of respective elements of the original second vector of operand values (it is well known in the art that a single vector/array value can be added to multiple elements of another vector/array. For example, see Cohoon et al., Page 493, Program 10.2) corresponding to elements of the first vector register of addressing values having identical values (In correcting errors in a vector write, it would not make sense to operate on values that were already correct, therefore only the incorrect (identical/duplicate) values should be modified .

11. As per Claim 4, Beard teaches: The method of claim 1, wherein address values for the sequence of addressed locations within the constrained area of memory are each calculated using a truncated portion of each respective addressing value of the first vector of addressing values (Column 7 Lines 30-36. Bits 6-0 (which are truncated from the 32-bit value) are used to index into memory).

12. As per Claim 7, Cray teaches: The method of claim 1, wherein for the loading of the third vector register with elements from memory, elements are loaded from locations specified by addressing values corresponding to bits of the bit vector that indicated a compare and no elements are loaded from locations specified by addressing values corresponding to bits of the bit vector that indicated a miscompare (Column 3, Lines 5-12. Beard teaches that a vector is loaded based on the addressing values of the first addressing vector register, but does not teach that the elements are loaded only from elements which indicated a compare. However, looking at the cmpress() function from Cray as described in Section 2.5.4, when the first addressing register is masked, only elements that have been indicated as a compare (if mj[I]) are stored in the destination register (the masked addressing register), and the others will be set as undefined, thus will be unable to load values).

13. As per Claim 9, Beard teaches: The method of claim 1, further comprising:

performing a first synchronization operation that ensures that the comparing the first sequence of values to the second sequence of values to generate the bit vector representing compares and miscompares effectively completes before the loading of the third vector register with elements from memory (inherent that Beards machine must compare before loading the third vector register, as loading the vector register is dependant upon the compare results, as seen in Claim 1 rejection); and

performing a first synchronization operation that ensures that the storing the result vector to memory completes before subsequent passes through a loop (inherent that in a non-bypassing, in-order processor, that all results must commit before a depending instruction can run, to ensure consistency. There is no evidence in the specification that the method is designed to run speculatively or out of order, and as such cannot execute instructions before a previous block is complete. See Patterson et al., Pages 195-196 for further information on why in-order retirement (and without speculation, in-order execution) is required in modern processors).

14. As per Claim 10, Beard teaches: A computer-readable medium having instructions stored thereon for causing a suitably programmed information-processing system to execute a method comprising:

providing a first vector of addressing values (Column 2 Line 65 – Column 3 Line 1);

providing a second vector of operand values (Column 3, Lines 10-12. The second vector instruction would necessarily need a vector of operand values in order to operate on the retrieved data words);

performing an arithmetic-logical operation using values from the third vector register and the compressed second vector of operand values to generate a result vector (Column 3, Lines 10-18); and

using the first vector of addressing values as masked by the bit vector, storing the result vector to memory (Column 3, Lines 13-18), but fails to teach:

storing a first sequence of values to a sequence of addressed locations within a constrained area of memory, wherein each location's address is based at least in part on a corresponding one of the addressing values;

reading back from the sequence of addressed locations values resulting from the storing of the first sequence to obtain a second sequence of values;

comparing the first sequence of values to the second sequence of values to generate a bit vector representing compares and miscompares.

However, Kohn has taught that in the prior art (the "conventional" algorithm as described on pages 4 and 5 of the specification) that after loading from memory, you would store a known pattern, load it back, and compare against the original pattern to determine collisions. Kohn further teaches that by doing this, not only would you see if there was a collision, but also which elements were colliding (Page 5). Given this disclosure, it would have been obvious to one of ordinary skill in the art at the time the invention was made to store a sequence of values to a constrained area of memory (which can be any writable area of memory for example, constrained can be read to be all of memory, or any arbitrary boundary of memory), reading it back out, and comparing the two to determine which elements did not compare correctly (and to do so, would require it to be stored somewhere), to allow correct execution of the instructions, which is critical for computers.

Beard further fails to teach:

compressing the second vector of operand values using the bit vector;

using the first vector of addressing values as masked by the bit vector, loading a third vector register with elements from memory.

However, Cray teaches an instruction set that runs vector instructions and supports a vector architecture, one of which would have been obvious to run on Beards invention, as Beard's invention is an improvement of the original Cray vector processor, as disclosed in Column 1 Line 66- Column 2, Line 6) and thus would be compatible, as every system needs some instruction set architecture, and Beard does not teach a specific instruction set to use in his invention. One of the instructions Cray teaches is a vector compress (Section 2.5.4), which uses a vector as a bit mask to modify a register to: invalidate certain elements, modify elements (if the destination is the same as source), and can also load a new register with elements (if the destination is different than the source), which is advantageous for improving the performance of decoupled execution, as also described in Section 2.5.4. Therefore, one of ordinary skill in the art at the time the invention was made would have recognized the improvement of decoupled execution of using mask and compression instructions, and would have modified Beards invention to include and utilize them to gain those performance advantages.

15. As per Claim 11, Beard teaches: A computerized method comprising:

loading a first vector register with addressing values (Column 2 Line 65 – Column 3 Line 1);

loading a second vector register with operand values (Column 3, Lines 10-12. The second vector instruction would necessarily need a vector of operand values in order to operate on the retrieved data words);

using at least some of the first vector register of addressing values, loading a third vector register with elements from memory (Column 3, Lines 1-10).

performing the arithmetic-logical operation using values from the third vector register and the combined second vector of operand values to generate a result vector (Column 3, Lines 13-18); and

using the at least some of the first vector register of addressing values, storing the result vector to memory (Column 3, Lines 13-18), but fails to teach:

storing a first sequence of values to a sequence of addressed locations within a constrained area of memory, wherein each one of these location's addresses in the constrained area of memory is based at least in part on a subset of bits of a corresponding one of the addressing values;

reading back from the sequence of addressed locations values resulting from the storing of the first sequence to obtain a second sequence of values;

comparing the first sequence of values to the second sequence of values.

However, Kohn has taught that in the prior art (the "conventional" algorithm as described on pages 4 and 5 of the specification) that after loading from memory, you would store a known pattern, load it back, and compare against the original pattern to determine collisions. Kohn further teaches that by doing this, not only would you see if there was a collision, but also which elements were colliding (Page 5). Given this disclosure, it would have been obvious to one of ordinary skill in the art at the time the invention was made to store a sequence of values to a constrained area of memory (which can be any writable area of memory for example, constrained can be read to be all of memory, or any arbitrary boundary of memory), reading it back out, and comparing the two to determine which elements did not compare correctly (and to do so, would require it to be stored somewhere), to allow correct execution of the instructions, which is critical for computers.

Beard further fails to teach:

selectively combining, with an arithmetic-logical operation, certain elements of the second vector of operand values based on results of the comparing;

However, Cray teaches an instruction set that runs vector instructions and supports a vector architecture, one of which would have been obvious to run on Beards invention, as Beard's invention is an improvement of the original Cray vector processor, as disclosed in Column 1 Line 66- Column 2, Line 6) and thus would be compatible, as every system needs some instruction set architecture, and Beard does not teach a specific instruction set to use in his invention. One of the instructions Cray teaches is a vector compress (Section 2.5.4), which uses a vector as a bit mask to modify a register to: invalidate certain elements, modify elements (if the destination is the same as source), and can also load a new register with elements (if the destination is different than the source), which is advantageous for improving the performance of decoupled execution, as also described in Section 2.5.4. Beard also teaches that an example arithmetic instruction that could be run on a vector is an add (Column 5, Lines 3-10). When a vector register is masked, and added to itself, then it has elements which are combined based on results of the comparing (when the comparison results are used as the mask). Therefore, one of ordinary skill in the art at the time the invention was made would have recognized the improvement of decoupled execution of using mask and compression instructions, and would have modified Beards invention to include and utilize them to gain those performance advantages.

16. As per Claim 12, Beard teaches: The method of claim 11, wherein addresses of the elements from memory are calculated by adding each respective addressing value to a base address (Column 3 Lines 2-5).

17. As per Claim 13, Beard teaches: The method of claim 11, wherein addresses of the elements from memory are calculated by performing a signed-addition operation of each respective addressing value to a base address of an object in memory (Column 6, Lines 48-49, which discloses the operands would be signed).

18. As per Claim 14, Beard teaches: The method of claim 11, wherein the arithmetic-logical operation is an addition operation that produces at least one element of the result vector as a summation of an element of the loaded third vector register and a

plurality of respective elements of the original second vector of operand values (it is well known in the art that a single vector/array value can be added to multiple elements of another vector/array. For example, see Cohoon et al., Page 493, Program 10.2) corresponding to elements of the first vector register of addressing values having identical values (In correcting errors in a vector write, it would not make sense to operate on values that were already correct, therefore only the incorrect (identical/duplicate) values should be modified.

19. As per Claim 15, Beard teaches: The method of claim 11, wherein address values for the sequence of addressed locations within the constrained area of memory are each calculated using a truncated portion of each respective addressing value of the first vector register of addressing values (Column 7 Lines 30-36. Bits 6-0 (which are truncated from the 32-bit value) are used to index into memory).

20. As per Claim 18, Cray teaches: The method of claim 11, wherein for the loading of the third vector register with elements from memory, elements are loaded from locations specified by addressing values corresponding to indications that indicated compares and no elements are loaded from locations specified by addressing values corresponding to indications that indicated miscompares (Column 3, Lines 5-12. Beard teaches that a vector is loaded based on the addressing values of the first addressing vector register, but does not teach that the elements are loaded only from elements which indicated a compare. However, looking at the cmpress() function from Cray as

described in Section 2.5.4, when the first addressing register is masked, only elements that have been indicated as a compare (if mj[I]) are stored in the destination register (the masked addressing register), and the others will be set as undefined, thus will be unable to load values).

21. As per Claim 19, Beard teaches: A computer-readable medium having instructions stored thereon for causing a suitably programmed information-processing system to execute the method of claim 11 (It is inherent that Beard's invention would be run off instructions stored on a computer-readable medium, otherwise it would not be possible for the computer system to execute the method).

22. As per Claim 21, Beard teaches: A system comprising:

a first vector processor having:

a first vector register having addressing values (Column 2 Line 65 – Column 3 Line 1);

a second vector register having operand values (Column 3, Lines 10-12. The second vector instruction would necessarily need a vector of operand values in order to operate on the retrieved data words);

a third vector register (Column 2, Lines 62-63, there are a plurality of registers);

a bit vector register (Column 2, Lines 62-63, a bit vector register is a register that holds more than 1 bit, which by definition would be a vector register);

circuitry that selectively performs an arithmetic-logical operation on corresponding values from the third vector register and the compressed second vector of operand values to generate values of a result vector (Column 3, Lines 10-18); and;

circuitry that selectively stores the result vector to memory (Column 3, Lines 13-18), but fails to teach:

circuitry that selectively stores a first sequence of values to a sequence of addressed locations within a constrained area of memory,

wherein each location's address is based at least in part on a corresponding one of the addressing values;

circuitry that selectively loads, from the sequence of addressed locations, values resulting from the stores of the first sequence to obtain a second sequence of values;

circuitry that selectively compares the first sequence of values to the second sequence of values to generate bit values into the bit vector register representing compares and miscompares.

However, Kohn has taught that in the prior art (the "conventional" algorithm as described on pages 4 and 5 of the specification) that after loading from memory, you would store a known pattern, load it back, and compare against the original pattern to determine collisions. Kohn further teaches that by doing this, not only would you see if there was a collision, but also which elements were colliding (Page 5). Given this disclosure, it would have been obvious to one of ordinary skill in the art at the time the invention was made to store a sequence of values to a constrained area of memory (which can be any writable area of memory for example, constrained can be read to be all of memory, or any arbitrary boundary of memory), reading it back out, and comparing the two to determine which elements did not compare correctly (and to do so, would require it to be stored somewhere), to allow correct execution of the instructions, which is critical for computers.

Beard further fails to teach:

circuitry that selectively compresses the second vector of operand values using the values in the bit vector register;

circuitry that selectively loads the third vector register with elements from memory addresses generated from the first vector register of addressing values as masked by the bit vector register.

However, Cray teaches an instruction set that runs vector instructions and supports a vector architecture, one of which would have been obvious to run on Beards invention, as Beard's invention is an improvement of the original Cray vector processor, as disclosed in Column 1 Line 66- Column 2, Line 6) and thus would be compatible, as every system needs some instruction set architecture, and Beard does not teach a specific instruction set to use in his invention. One of the instructions Cray teaches is a vector compress (Section 2.5.4), which uses a vector as a bit mask to modify a register to: invalidate certain elements, modify elements (if the destination is the same as source), and can also load a new register with elements (if the destination is different than the source), which is advantageous for improving the performance of decoupled execution, as also described in Section 2.5.4. Since there are instructions to perform these operations, the circuitry to perform them must also exist in the system. Therefore, one of ordinary skill in the art at the time the invention was made would have recognized the improvement of decoupled execution of using mask and compression instructions, and would have modified Beards invention to include and utilize them to gain those performance advantages, along with the circuitry required to use them.

23. As per Claim 22, Beard teaches: The system of claim 21, further comprising circuitry to calculate addresses of the elements in memory by adding each respective addressing value to a base address value (Column 3 Lines 2-5).

24. As per Claim 23, Beard teaches: The system of claim 21, wherein the arithmetic-logical operation is an addition operation that produces at least one element of the result vector as a summation of an element of the loaded third vector register and a plurality of respective elements of the original second vector of operand values (it is well known in the art that a single vector/array value can be added to multiple elements of another vector/array. For example, see Cohoon et al., Page 493, Program 10.2) corresponding to elements of the first vector register of addressing values having identical values (In correcting errors in a vector write, it would not make sense to operate on values that were already correct, therefore only the incorrect (identical/duplicate) values should be modified.

25. As per Claim 24, Beard teaches: The system of claim 21, further comprising

circuitry to calculate address values for the sequence of addressed locations within the constrained area of memory using a truncated portion of each respective addressing value of the first vector register of addressing values (Column 7 Lines 30-36. Bits 6-0 (which are truncated from the 32-bit value) are used to index into memory).

26. As per Claim 27, Cray teaches: The system of claim 21, wherein the circuitry that selectively loads the third vector register with elements from memory only loads element from locations specified by addressing values corresponding to bits of the bit vector that indicated a compare (Column 3, Lines 5-12. Beard teaches that a vector is loaded based on the addressing values of the first addressing vector register, but does not teach that the elements are loaded only from elements which indicated a compare. However, looking at the cmpress() function from Cray as described in Section 2.5.4, when the first addressing register is masked, only elements that have been indicated as a compare (if mj[I]) are stored in the destination register (the masked addressing register), and the others will be set as undefined, thus will be unable to load values).

27. As per Claim 28, Beard teaches: The system of claim 21, further comprising:

synchronization circuitry that ensures that the comparing the first sequence of values to the second sequence of values to generate the bit vector representing compares and miscompares effectively completes before the loading of the third vector register with elements from memory (inherent that Beards machine must compare before loading the third vector register, as loading the vector register is dependant upon the compare results, as seen in Claim 1 rejection), and that ensures that the storing the result vector to memory completes before subsequent passes through a loop (inherent that in a non-bypassing, in-order processor, that all results must commit before a depending instruction can run, to ensure consistency. There is no evidence in the specification that the method is designed to run speculatively or out of order, and as such cannot execute instructions before a previous block is complete. See Patterson et al., Pages 195-196 for further information on why in-order retirement (and without speculation, in-order execution) is required in modern processors).

28. As per Claim 29, Beard teaches: The system of claim 21, further comprising:

a second vector processor having:

a first vector register having addressing values;

a second vector register having operand values;

a third vector register;

a bit vector register;

circuitry that selectively stores a first sequence of values to a sequence of addressed locations within a constrained area of memory, wherein each location's address is based at least in part on a corresponding one of the addressing values;

circuitry that selectively loads, from the sequence of addressed locations, values resulting from the stores of the first sequence to obtain a second sequence of values;

circuitry that selectively compares the first sequence of values to the second sequence of values to generate bit values into the bit vector register representing compares and miscompares;

circuitry that selectively compresses the second vector of operand values using the values in the bit vector register;

circuitry that selectively loads the third vector register with elements from memory addresses generated from the first vector register of addressing values as masked by the bit vector register.

circuitry that selectively performs an arithmetic-logical operation on corresponding values from the third vector register and the compressed second vector of operand values to generate values of a result; and;

circuitry that selectively stores the result vector to memory (See Claim 21 rejection for this and all of the above); and

synchronization circuitry that ensures that the comparing the first sequence of values to the second sequence of values to generate the bit vector representing compares and miscompares effectively completes in both the first and second vector processors before the loading of the third vector register with elements from memory in either processor (inherent that Beards machine must compare before loading the third vector register, as loading the vector register is dependant upon the compare results, as seen in Claim 21 rejection), and that ensures that the storing the result vector to memory completes before subsequent passes through a loop (inherent that in a non-bypassing, in-order processor, that all results must commit before a depending instruction can run, to ensure consistency. There is no evidence in the specification that the method is designed to run speculatively or out of order, and as such cannot execute instructions before a previous block is complete. See Patterson et al., Pages 195-196 for further information on why in-order retirement (and without speculation, in-order execution) is required in modern processors).

29. As per Claim 30, Beard teaches: A system comprising:

a first vector register (Column 2, Lines 62-63, there are a plurality of registers);

a second vector register (Column 2, Lines 62-63, there are a plurality of registers);

a third vector register (Column 2, Lines 62-63, there are a plurality of registers);

a bit vector register (Column 2, Lines 62-63, there are a plurality of registers);

means for loading the first vector register with addressing values (Column 2 Line 65 – Column 3 Line 1);

means for loading the second vector register with operand values (Column 3, Lines 10-12. The second vector instruction would necessarily need a vector of operand values in order to operate on the retrieved data words);

means for loading a third vector register with elements from memory address locations generated using at least some of the first vector register of addressing values (Column 3, Lines 1-10).

means for performing the arithmetic-logical operation using values from the third vector register and the combined second vector of operand values to generate a result vector (Column 3, Lines 10-18); and

means for storing the result vector to memory (Column 3, Lines 10-18), but fails to teach:

means for storing a first sequence of values to a sequence of addressed locations within a constrained area of memory, wherein each one of these location's addresses in the constrained area of memory is based at least in part on a subset of bits of a corresponding one of the addressing values;

means for loading from the sequence of addressed locations values resulting from the storing of the first sequence to obtain a second sequence of values;

means for comparing the first sequence of values to the second sequence of values.

However, Kohn has taught that in the prior art (the "conventional" algorithm as described on pages 4 and 5 of the specification) that after loading from memory, you would store a known pattern, load it back, and compare against the original pattern to determine collisions. Kohn further teaches that by doing this, not only would you see if there was a collision, but also which elements were colliding (Page 5). Given this disclosure, it would have been obvious to one of ordinary skill in the art at the time the invention was made to store a sequence of values to a constrained area of memory (which can be any writable area of memory for example, constrained can be read to be all of memory, or any arbitrary boundary of memory), reading it back out, and comparing the two to determine which elements did not compare correctly (and to do so, would require it to be stored somewhere), to allow correct execution of the instructions, which is critical for computers.

Beard further fails to teach:

means for selectively combining, with an arithmetic-logical operation, certain elements of the second vector of operand values based on results of the comparing.

However, Cray teaches an instruction set that runs vector instructions and supports a vector architecture, one of which would have been obvious to run on Beards invention, as Beard's invention is an improvement of the original Cray vector processor, as disclosed in Column 1 Line 66- Column 2, Line 6) and thus would be compatible, as every system needs some instruction set architecture, and Beard does not teach a specific instruction set to use in his invention. One of the instructions Cray teaches is a vector compress (Section 2.5.4), which uses a vector as a bit mask to modify a register to: invalidate certain elements, modify elements (if the destination is the same as source), and can also load a new register with elements (if the destination is different than the source), which is advantageous for improving the performance of decoupled execution, as also described in Section 2.5.4. Beard also teaches that an example arithmetic instruction that could be run on a vector is an add (Column 5, Lines 3-10). When a vector register is masked, and added to itself, then it has elements which are combined based on results of the comparing (when the comparison results are used as the mask). Therefore, one of ordinary skill in the art at the time the invention was made would have recognized the improvement of decoupled execution of using mask and compression instructions, and would have modified Beards invention to include and utilize them to gain those performance advantages.

30. As per Claim 31, Beard teaches: A system comprising:

a first vector register that can be loaded with addressing values (Column 2 Line 65 – Column 3 Line 1) ;

a second vector register that can be loaded with operand values (Column 3, Lines 10-12. The second vector instruction would necessarily need a vector of operand values in order to operate on the retrieved data words);

a third vector register that can be loaded with operand values from memory locations indirectly addressed using the addressing values from the first vector register (Column 3, Lines 19-22, while Column 28, Lines 18-19 show indirect addressing capabilities);

a circuit that uses indirect addressing to selectively load the third vector register with elements from memory (Column 3 Lines 5-10, while Column 28, Lines 18-19 show indirect addressing capabilities);

a circuit that selectively adds values from the third vector register and the second vector of operand values to generate a result vector (Column 3, Lines 10-18); and

a circuit that selectively stores the result vector to memory using indirect addressing (Column 3, Lines 11-18, while Column 28, Lines 18-19 show indirect addressing capabilities), but fails to teach:

a circuit that determines elements of the first vector register that have an address value that duplicates a value in another element address.

However, Kohn has taught that in the prior art (the "conventional" algorithm as described on pages 4 and 5 of the specification) that after loading from memory, you would store a known pattern, load it back, and compare against the original pattern to determine collisions. Kohn further teaches that by doing this, not only would you see if there was a collision, but also which elements were colliding (Page 5). Given this disclosure, it would have been obvious to one of ordinary skill in the art at the time the invention was made to store a sequence of values to memory, read it back out, and comparing the two to determine which elements did not compare correctly (and to do so, would require a circuit to do so), to allow correct execution of the instructions, which is critical for computers.

Beard further fails to teach:

a circuit that selectively adds certain elements of the second vector of operand values based on the elements having the duplicated address values.

However, Cray teaches an instruction set that runs vector instructions and supports a vector architecture, one of which would have been obvious to run on Beards invention, as Beard's invention is an improvement of the original Cray vector processor, as disclosed in Column 1 Line 66- Column 2, Line 6) and thus would be compatible, as every system needs some instruction set architecture, and Beard does not teach a specific instruction set to use in his invention. One of the instructions Cray teaches is a vector compress (Section 2.5.4), which uses a vector as a bit mask to modify a register to: invalidate certain elements, modify elements (if the destination is the same as source), and can also load a new register with elements (if the destination is different than the source), which is advantageous for improving the performance of decoupled execution, as also described in Section 2.5.4. Beard also teaches that an example arithmetic instruction that could be run on a vector is an add (Column 5, Lines 3-10). When a vector register is masked, and added to itself, then it has elements which are combined based on results of the comparing (when the comparison results are used as the mask). Therefore, one of ordinary skill in the art at the time the invention was made would have recognized the improvement of decoupled execution of using mask and compression instructions, and would have modified Beards invention to include and utilize them to gain those performance advantages.

31. As per Claim 32, Beard teaches: The system of claim 31, further comprising:

an adder that generates addresses of the elements from memory by adding each respective addressing value to a base address (Column 3 Lines 2-5).

32. As per Claim 33, Beard teaches: The system of claim 31, further comprising:

an adder that generates addresses of the elements from memory by a signed-addition operation of each respective addressing value to a base address of an object in memory (Column 6, Lines 48-49, which discloses the operands would be signed).

33. As per Claim 34, Beard teaches: The system of claim 31, wherein the circuit that selectively adds certain elements performs one or more addition operations using those values from a plurality of respective elements of the original second vector of operand values (it is well known in the art that a single vector/array value can be added to multiple elements of another vector/array. For example, see Cohoon et al., Page 493, Program 10.2) corresponding to elements of the first vector register of addressing values having identical values (In correcting errors in a vector write, it would not make sense to operate on values that were already correct, therefore only the incorrect (identical/duplicate) values should be modified).

Response to Arguments

34. Examiner appreciates Applicants new declaration to show actual reduction to practice of the invention, prior to the Cray reference. However, there is just a little more that Examiner would like to see before withdrawing the rejection of the claims. To quote the MPEP, Section 715.07(III), "proof of actual reduction to practice requires a showing that the apparatus actually existed and worked for its intended purpose". While the attached email with the code shows that there was at least the concept of the invention prior to the date, Examiner is not sure it by itself is sufficient to show actual reduction to practice. The email shows that the algorithm is not yet in the compiler, and while it is stated that there is a 4x speedup over the previous algorithm with the code, it is not clear if the code has actually been run and tested, or if it is just an expected result. Additionally, as the code appears to be in machine language, there is no way for the

Examiner to actually verify that the code does implement the claimed invention, although the comments seem to indicate that it does.

What Examiner would like to see as evidence, which would, in the Examiners eyes, show actual reduction to practice, is a test result of this code, showing that the code show not only implements the claimed invention, but also shows that the code worked at the time this email was sent (or at any time prior to the reference date). If Examiner can see evidence that the code as disclosed in the email not only performs correctly, yielding the 4x speedup, but also that the code performs the claimed invention (which the test results should also show), then Examiner will agree that an actual reduction to practice was performed prior to the reference date, and withdraw the Cray reference from the rejections.

35. Regarding Applicants arguments towards the 103 rejections, Examiner believes that the proposed combination of references would teach the claim limitations, as Kohn and Cray teach the elements of the claims that the other reference lacks, which would result in the claimed invention (Cray teaches the elements Applicant says Kohn does not teach, and vice versa).

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Robert E. Fennema whose telephone number is (571) 272-2748. The examiner can normally be reached on Monday-Friday, 8:45-6:15.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Robert E Fennema
Examiner
Art Unit 2183

[illegible]

RICHARD L. ELLIS
PRIMARY EXAMINER

RF